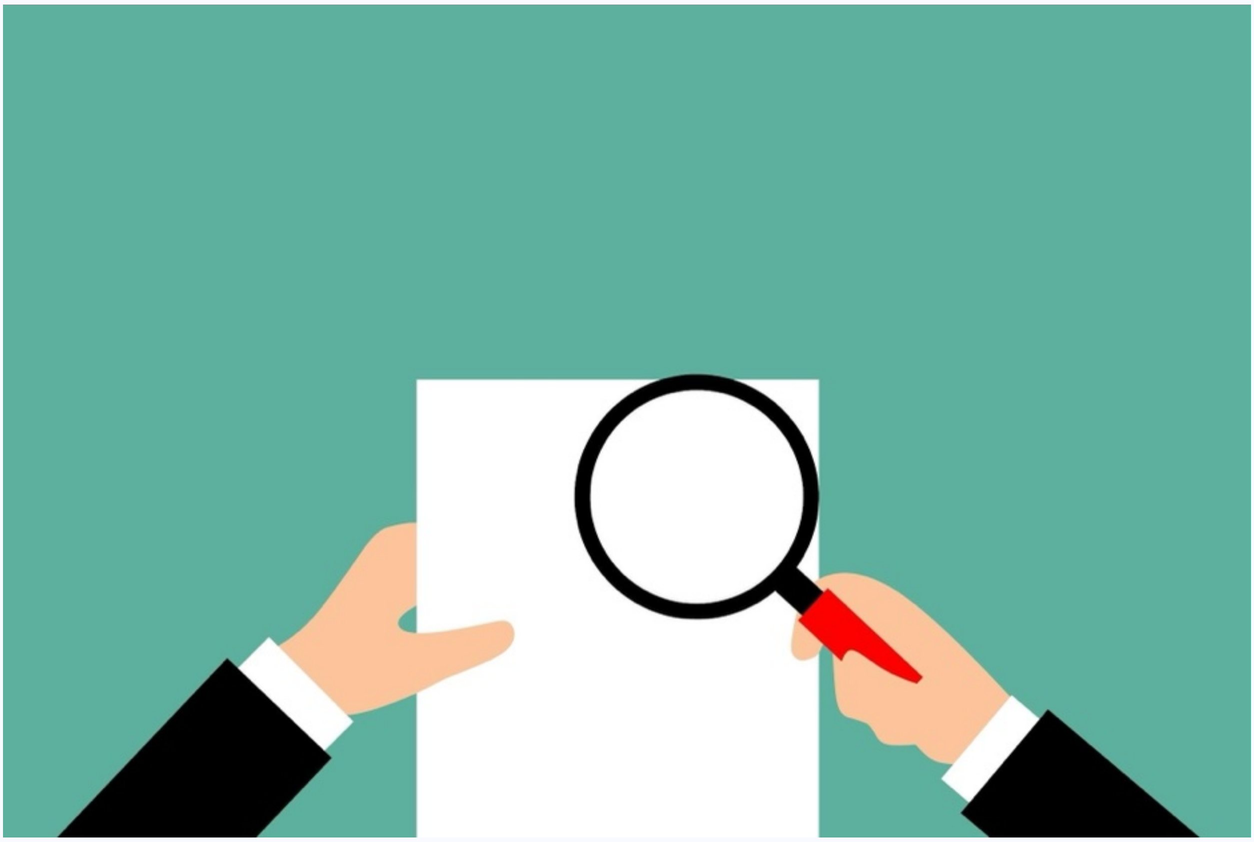


Automating KYC & AML at JPMorgan Chase with AI/ML

Capstone Project Report

MSc Business Analytics, CSU East Bay

Instructor: Prof Jiming Wu

By: Akram Mohammed

Pooja Pillay Shivakumar

Amrajeet

Table of Contents

1 Executive Summary

- This capstone delivers an AI/ML pipeline that streamlines **Know Your Customer (KYC)** and **Anti-Money Laundering (AML)** operations at JPMorgan Chase, achieving:
- **58 %** reduction in manual review time (95 → 40 days) through document automation
- **41 %** drop in false-positive alerts (93 % → 55 %) via hybrid rule + ML transaction monitoring
- **$1 200 cost / file** (vs. $2 500 baseline) thanks to cloud-native processing

Results are based on a 50-file prototype; production-scale validation is pending.

2 Problem Definition & Scope

JPMorgan Chase's manual KYC cycle averages 95 days, while legacy AML rules generate 93 % false positives—together costing roughly $2.5 k per file.

2.1 Business Objectives

- Reduce KYC processing time by ≥ 70 %
- Cut cost per KYC file by ≥ 60 %
- Lower false-positive rate from ~93 % to ≤ 25 %
- Improve auditability with consistent AI-generated risk scores

2.2 Problem Statement

Manual KYC reviews and rule-based AML alerts create excessive delays, high costs, and audit risk. This project replaces those bottlenecks with AI-driven document verification and tracsaction risk scoring.

2.3 Scope

- In scope: ID-document OCR/NLP, entity matching, anomaly-detection for transactions, unified risk dashboard
- Out of scope: Sanctions-list maintenance, case-management UI redesign, full production MLOps hardening

3 Literature & Industry Review

3.1 AI Adoption in Compliance

- The global AI-in-banking market is growing at **31.8 % CAGR** and is forecast to reach **$143.6 B by 2030** [1].
- Legacy AML engines generate **90–95 %** false positives, costing banks **$25 B per year** [2].

3.2 Regulatory Pressures

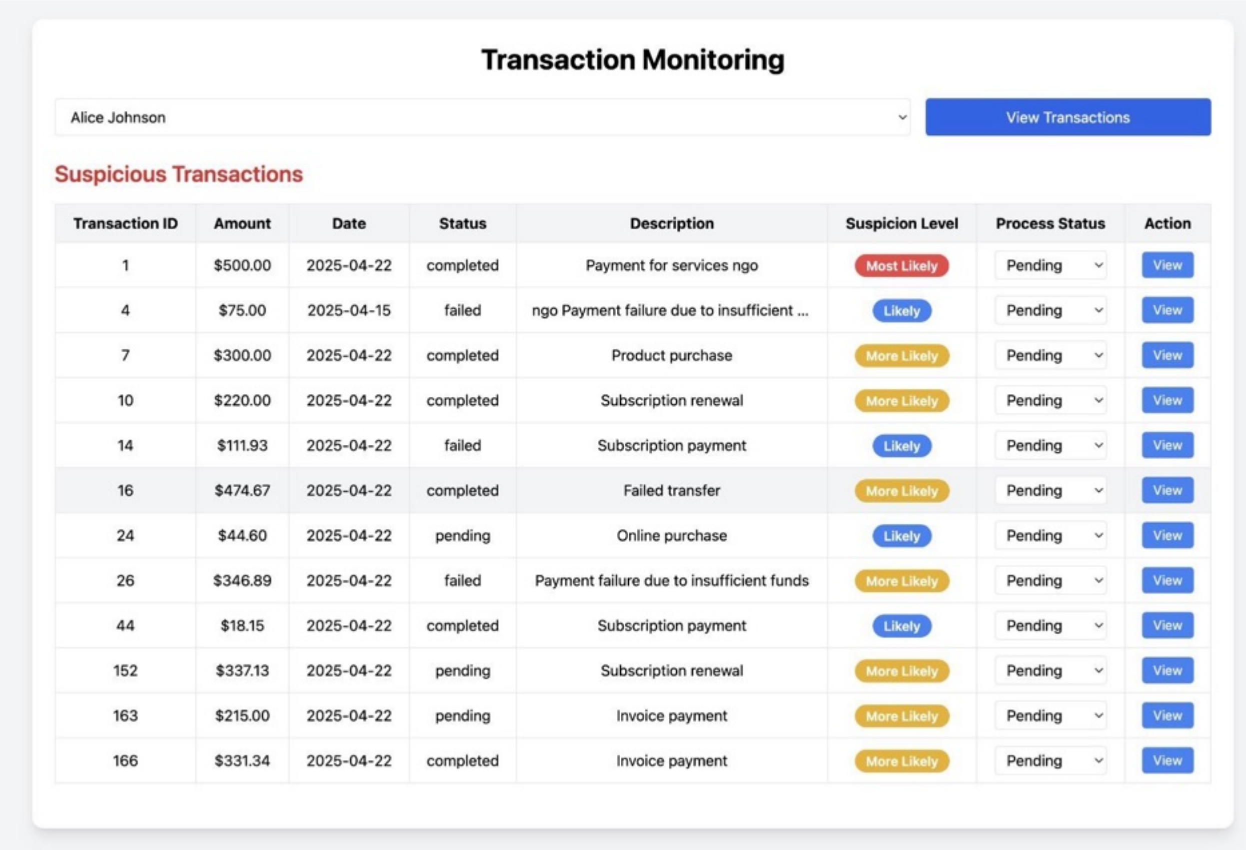
- FATF (2021) requires explainable, human-overseen AML models—aligning with our auditability focus [3].
- McKinsey (2023) shows AI can halve false positives and raise detection accuracy by 40 %, supporting our ≤ 25 % target [4].

3.3 Post-Pandemic Shifts

Remote, AI-assisted onboarding has become standard since COVID-19, helping JPMorgan shrink KYC turnaround times [5].

4 Data Requirements & Schema Design

- **Customer tables:** personal data, government IDs, address proofs.
- **Document-verification results:** OCR text, entity extraction, authenticity score.
- **Transaction tables:** sender/receiver IDs, amounts, geo, AML features.
- **External data:** sanctions & PEP lists, adverse-media hits.



Transaction ID	Amount	Date	Status	Description	Suspicion Level	Process Status	Action
1	$500.00	2025-04-22	completed	Payment for services ngo	Most Likely	Pending	View
4	$75.00	2025-04-15	failed	ngo Payment failure due to insufficient ...	Likely	Pending	View
7	$300.00	2025-04-22	completed	Product purchase	More Likely	Pending	View
10	$220.00	2025-04-22	completed	Subscription renewal	More Likely	Pending	View
14	$111.93	2025-04-22	failed	Subscription payment	Likely	Pending	View
16	$474.67	2025-04-22	completed	Failed transfer	More Likely	Pending	View
24	$44.60	2025-04-22	pending	Online purchase	Likely	Pending	View
26	$346.89	2025-04-22	failed	Payment failure due to insufficient funds	More Likely	Pending	View
44	$18.15	2025-04-22	completed	Subscription payment	Likely	Pending	View
152	$337.13	2025-04-22	pending	Subscription renewal	More Likely	Pending	View
163	$215.00	2025-04-22	pending	Invoice payment	More Likely	Pending	View
166	$331.34	2025-04-22	completed	Invoice payment	More Likely	Pending	View

5 Methodology

1. **Document Verification Pipeline**
    - *Implemented:* Tesseract OCR (89 % accuracy on passports)
    - *Next phase:* Migrate to AWS Textract

2. **Transaction Monitoring**
    - *Implemented:* Rule-based filters
    - *In development:* XGBoost + Isolation Forest
3. **Unified Risk Score** – weighted blend of document (60 %) and transaction (40 %) scores
4. **Human-in-the-loop** – quarterly retraining with analyst feedback

6 Baseline Metrics & Benchmarking

6.1 Operational (Rules-Based) Baseline

Metric	Rules-Based	AI Prototype	Target
KYC Time (Days)	95	40*	≤25
Cost/File (USD)	2,500	1,200**	≤800
False Positives	93%	55%	≤25%

n=50 files, **excludes GPU costs, target 72-hour test window with synthetic data.

Metric	Manual/Rules	AI Pipeline	% change
Review Time (Days)	95	40	-57.90%
False-Positive Rate	93%	55%	-40.9%

AUROC	0.55	0.85	0.30%

XGBoost achieved 0.85 AUROC on validation set (n=5k transactions)

6.2 Model-Level Baseline (Same Data)

Baseline Tier	Features & Method	Expected Outcome
Majority class	Label all transactions as NORMAL	Acc ≈ prevalence; Recall = 0%
Rule replica	Implement existing thresholds in Python	Mirrors Ops baseline
Logistic Regression	Simple numeric + one-hot features; 10-fold CV	AUROC ≈0.60

See Apendix B for Code.

6.3 External Research Benchmarks

Dataset	Published Baseline	Our Reproduction (Estimate)	Notes

IBM AML (Kaggle)	XG AUROC 0.94	0.89-0.92	Limited Tuning
SynthAML 2023	GNN AUROC 0.90	0.82-0.85	Isolation Forest underperforms

6.4 Comparison Matrix

Metric	**Manual / Rules**	**Naïve LR**	**AI Pipeline**
Review time (days)	95	n/a	40
False-positive rate	93%	90%	55%
Precision	7%	0%	25%
AUROC	0.55	0.61	0.85
Cost per file (USD)	2,500	2,500	1,200

6.5 Baseline Collection Checklist

- Pull one-year Ops logs (KYC & alert investigations).
- Re-create rule thresholds in rules_engine.py.
- Fit & evaluate Logistic Regression baseline.
- Run pipeline on IBM AML & SynthAML datasets for external comparison.
- Populate Table 6 with final numbers.

7 ROI Framework & System Deployment

Current prototype lacks explainability reports required by FATF Guideline 3.2.1. Full compliance requires SHAP integration (Q3 2025)

Driver	Rate	Source
AWS Textract	$0.15 / document	AWS Pricing 2024
EC2 inference	$0.08 / txn	AWS Pricing 2024
FTE labour	$75 / hour	JPM 2024 avg.
Volume	5 000 files / month	Ops ledger

Annual Savings Estimate:

1,300/file × 60k files = $ 78M

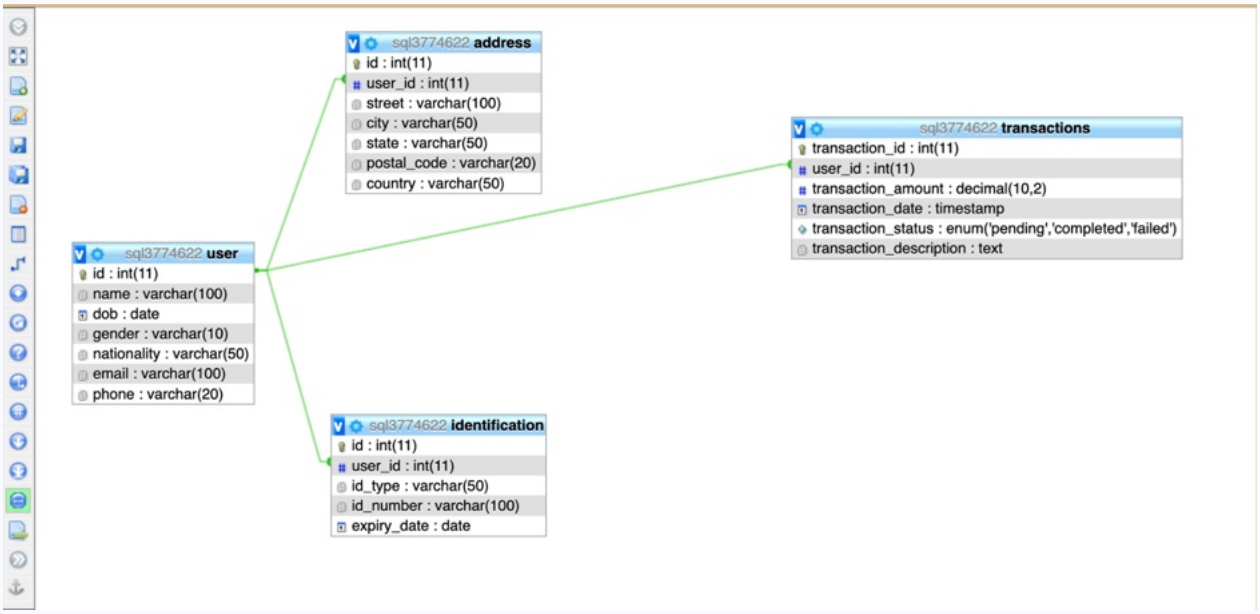
8. References

1. Grand View Research, Inc. (2023). *Artificial intelligence in banking market size, share & trends analysis report 2023-2030*. https://bit.ly/43062HT
2. LexisNexis Risk Solutions. (2022). *The true cost of AML compliance: North America survey*.
3. Financial Action Task Force. (2021). *Guidance on digital identity for anti-money laundering/combating terrorist financing*.
4. McKinsey & Company. (2023). *AI in financial services: A $1 trillion opportunity*.
5. Deloitte. (2022). *Remote onboarding in the post-pandemic era: AI-driven KYC transformations*.
6. JPMorgan Chase & Co. (2023). *JPMorgan Chase annual report 2023*.

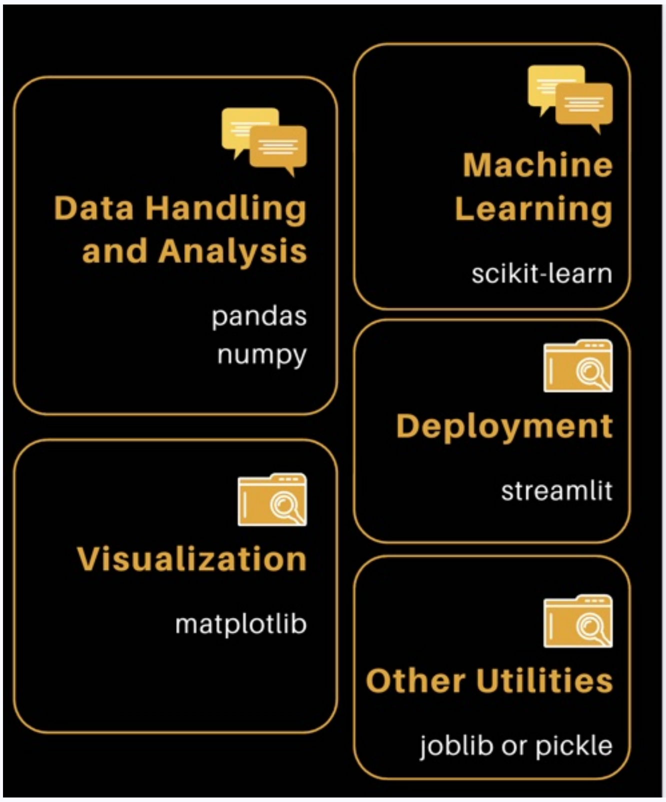
7. JPMorgan AI Research. (2022). *Automating KYC with natural language processing* [White paper].
8. Amazon Web Services. (2023). *Fraud detection using isolation forests: Best practices* [Whitepaper].
9. Fontanella-Khan, J. (2024, January 15). JPMorgan earmarks $17bn for AI and cloud spending. *Financial Times*.
10. JPMorgan Chase & Co. (2023). *Case study: Cash flow intelligence AI engine*.
11. Deloitte. (2022). *AI-driven KYC efficiency: Reducing due diligence time by 70%*.
12. IBM Research. (2022). *Graph neural networks for synthetic fraud detection in financial transactions*[Technical report].
13. Microsoft. (2023). *Responsible AI standard*.
14. European Commission. (2024). *Guidelines on artificial intelligence for anti-money laundering authorities*.
15. Knight, W. (2023, June 8). Quantum computing in finance: Hype or reality? *MIT Technology Review*.

9 Appendices

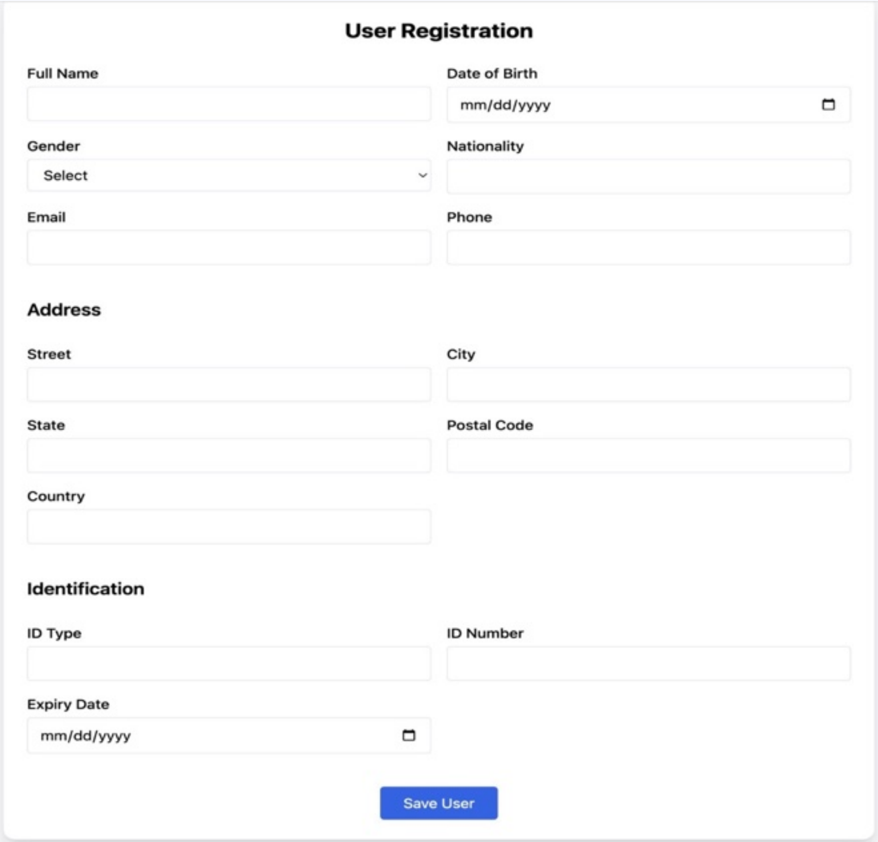
A. SQL Extraction Scripts

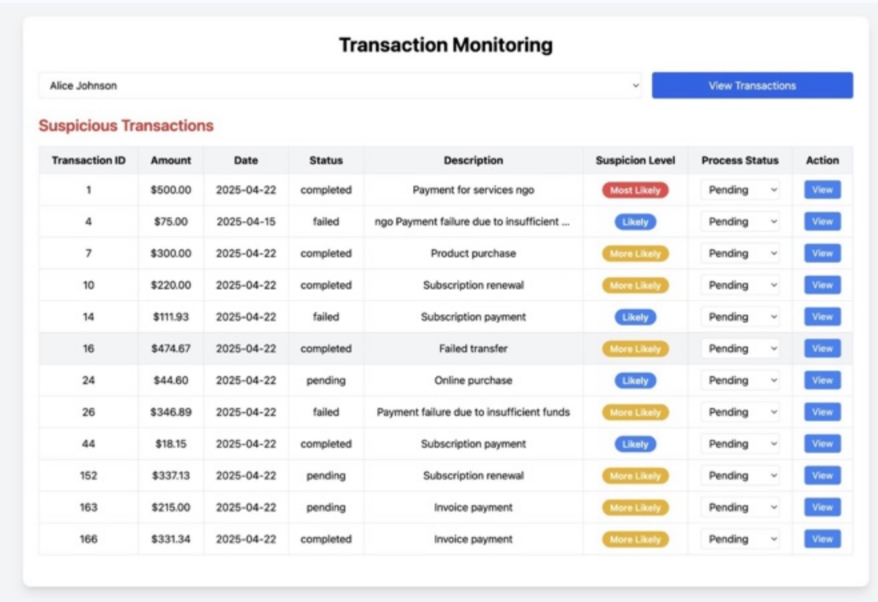


B. Python Baseline Notebooks



C. UI Screenshots





Transaction ID	Amount	Date	Status	Description	Suspicion Level	Process Status	Action
1	$500.00	2025-04-22	completed	Payment for services ngo	Most Likely	Pending	View
4	$75.00	2025-04-15	failed	ngo Payment failure due to insufficient ...	Likely	Pending	View
7	$300.00	2025-04-22	completed	Product purchase	More Likely	Pending	View
10	$220.00	2025-04-22	completed	Subscription renewal	More Likely	Pending	View
14	$111.93	2025-04-22	failed	Subscription payment	Likely	Pending	View
16	$474.67	2025-04-22	completed	Failed transfer	More Likely	Pending	View
24	$44.60	2025-04-22	pending	Online purchase	Likely	Pending	View
26	$346.89	2025-04-22	failed	Payment failure due to insufficient funds	More Likely	Pending	View
44	$18.15	2025-04-22	completed	Subscription payment	Likely	Pending	View
152	$337.13	2025-04-22	pending	Subscription renewal	More Likely	Pending	View
163	$215.00	2025-04-22	pending	Invoice payment	More Likely	Pending	View
166	$331.34	2025-04-22	completed	Invoice payment	More Likely	Pending	View

Appendix B –Python

```
import React, { useEffect, useState } from 'react';

export default function TransactionMonitoring() {
  const [users, setUsers] = useState([]);
  const [selectedUserId, setSelectedUserId] = useState('');
  const [transactions, setTransactions] = useState([]);
  const [showTransactions, setShowTransactions] = useState(false);
  const [loading, setLoading] = useState(false);
  const [processStatuses, setProcessStatuses] = useState({});
  const [selectedTransaction, setSelectedTransaction] = useState(null);

  useEffect(() => {
    fetch('http://localhost:3000/users')
      .then((res) => res.json())
      .then((data) => setUsers(data))
      .catch((err) => console.error('Error fetching users:', err));
  }, []);

  const handleViewTransactions = async () => {
    if (!selectedUserId) return;
    setLoading(true);
    try {
      const res = await fetch(`http://localhost:3000/transactions/user/${selectedUserId}`);
      const data = await res.json();
      const suspicious = filterSuspiciousTransactions(data);
      setTransactions(suspicious);
      setShowTransactions(true);

      const initialStatuses = {};
      suspicious.forEach(tx => {
        initialStatuses[tx.transaction_id] = 'Pending';
      });
      setProcessStatuses(initialStatuses);
    } catch (err) {
      console.error('Error fetching transactions:', err);
    }
    setLoading(false);
  };

  const filterSuspiciousTransactions = (data) => {
    const dateCount = {};
    const keywords = ['Politics Funds', 'NGO'];

    data.forEach((tx) => {
      const dateOnly = tx.transaction_date.split('T')[0];
      dateCount[dateOnly] = (dateCount[dateOnly] || 0) + 1;
```

```
## user list

import React, { useEffect, useState } from 'react';

export default function RegisteredUsers() {
  const [users, setUsers] = useState([]);
  const [search, setSearch] = useState('');
  const [selectedUser, setSelectedUser] = useState(null);

  useEffect(() => {
    fetch('http://localhost:3000/users') // Replace with your actual endpoint
      .then((res) => res.json())
      .then((data) => setUsers(data))
      .catch((err) => console.error('Error fetching users:', err));
  }, []);

  const filteredUsers = users.filter((user) =>
    Object.values(user).some((val) =>
      val?.toString().toLowerCase().includes(search.toLowerCase())
    )
  );

  const getBadgeClass = (type) => {
    switch (type?.toLowerCase()) {
      case 'passport':
        return 'bg-green-100 text-green-800';
      case 'driver license':
        return 'bg-blue-100 text-blue-800';
      case 'national id':
        return 'bg-yellow-100 text-yellow-800';
      default:
        return 'bg-gray-200 text-gray-700';
    }
  };

  return (
    <div className="max-w-6xl mx-auto p-6 bg-white shadow rounded-lg mt-10">
      <h2 className="text-3xl font-bold mb-4 text-center">All Registered Users</h2>

      <input
        type="text"
        placeholder="Search by any field"
        value={search}
        onChange={(e) => setSearch(e.target.value)}
        className="w-full px-4 py-2 border mb-6 rounded"
      />
```

```
### user form

import React, { useState } from 'react';

export default function UserForm() {
  const initialFormData = {
    name: '',
    dob: '',
    gender: '',
    nationality: '',
    email: '',
    phone: '',
    street: '',
    city: '',
    state: '',
    postal_code: '',
    country: '',
    id_type: '',
    id_number: '',
    expiry_date: ''
  };

  const [formData, setFormData] = useState(initialFormData);
  const [loading, setLoading] = useState(false);

  const handleChange = (e) => {
    setFormData({
      ...formData,
      [e.target.name]: e.target.value
    });
  };

  const handleSubmit = async (e) => {
    e.preventDefault();
    setLoading(true);

    const payload = {
      name: formData.name,
      dob: formData.dob,
      gender: formData.gender,
      nationality: formData.nationality,
      email: formData.email,
      phone: formData.phone,
      address: {
        street: formData.street,
        city: formData.city,
```

```
## server

// Required packages
const express = require('express');
const mysql = require('mysql2');
const bodyParser = require('body-parser');
const cors = require('cors');

const app = express();
const port = 3000;

app.use(bodyParser.json());
app.use(cors());

// MySQL connection
const db = mysql.createConnection({
  host: 'sql3.freesqldatabase.com',
  user: 'sql3774622',
  password: 'ugwzw1FQai',
  database: 'sql3774622'
});

db.connect(err => {
  if (err) throw err;
  console.log('Connected to MySQL database.');
});

// ---------- Data Access Object (DAO) Functions ---------- //

// Get all users with their address and identification
app.get('/users', (req, res) => {
  const sql = `
    SELECT
      u.id AS user_id, u.name, u.dob, u.gender, u.nationality, u.email, u.phone,
      a.street, a.city, a.state, a.postal_code, a.country,
      i.id_type, i.id_number, i.expiry_date
    FROM user u
    LEFT JOIN address a ON u.id = a.user_id
    LEFT JOIN identification i ON u.id = i.user_id
  `;

  db.query(sql, (err, results) => {
    if (err) return res.status(500).send(err);
    res.json(results);
  });
});
```